



A Robust Multi-Layer Stacking Model for Malicious Node Classification in Vehicular Networks

Kulla Swetha¹, P. Shireesha^{2*}, Chiluveru Dinesh¹, Nutenki Nagaraju¹, Nagulapalli Manikanta¹, N Nagaraj Reddy¹

¹UG Student, ²Assistant Professor, ^{1,2}Department of Computer Science and Engineering (AI&ML)

^{1,2}Vaagdevi Engineering College, Bollikunta, Warangal, 506005, Telangana, India

*Correspondence: P. Shireesha (psirisha780@gmail.com)

Abstract

Vehicular Ad Hoc Networks (VANETs) play a vital role in modern Intelligent Transportation Systems by enabling real-time communication between vehicles and infrastructure to enhance road safety and traffic efficiency. With the rapid advancement of 5G networks and the rise of connected and autonomous vehicles, VANETs have evolved into highly dynamic and data-intensive environments. Traditionally, security in these networks relied on rule-based mechanisms and signature-based Intrusion Detection Systems (IDS), which are no longer effective due to their inability to adapt to evolving attack patterns and large-scale data. The framework incorporates classifiers such as K-Nearest Neighbors (KNN), Support Vector Classifier (SVC), and Gaussian Naive Bayes (GNB), along with a hybrid ensemble model called Stacked Tao Tree (STT). Additionally, advanced data balancing methods, including Adaptive Synthetic Sampling (ADASYN) was employed to address class imbalance and improve learning performance. Experimental results demonstrate that the proposed STT model achieves exceptional accuracy, reaching 100.00% for denial-of-service attacks, 99.90% for sybil attack attempts, and 100.00% for blackhole attacks. The study provides a scalable and efficient solution for secure vehicular communication in dynamic network environments.

Keywords: Classification task, intrusion detection, machine learning, vehicular networks, wireless communication, XAI

1. Introduction

A VANET is a specialized form of Mobile Ad Hoc Network that enables wireless communication among vehicles and between vehicles and roadside infrastructure. Vehicle-to-Vehicle communication supports safety applications such as collision warnings and adaptive driving assistance, but it is highly exposed to malicious nodes that can disrupt communication and compromise road safety [1]. Likewise, Vehicle-to-Infrastructure communication facilitates interaction with traffic control systems and management centers, improving traffic flow and efficiency, yet it remains vulnerable to threats such as spoofing, eavesdropping, and distributed denial of service attacks due to its dependence on open wireless channels [2].

As a key component of Intelligent Transportation Systems, VANETs enable real-time information exchange that enhances traffic safety and operational efficiency through applications like emergency alerts and congestion management [3]. However, the decentralized and rapidly changing nature of these networks introduces significant security challenges. Frequent topology changes, high node mobility, and strict latency requirements make it difficult to maintain secure and reliable communication [4].

Among various security threats, distributed denial of service attacks is particularly harmful as they flood the network with excessive traffic, interrupting the delivery of critical safety messages [5]. This can lead to increased packet loss, communication delays, and potential hazards on the road. Conventional security solutions such as firewalls and signature-based IDS are not well-suited for such dynamic environments, as they lack adaptability and real-time responsiveness. Consequently, more advanced,



and intelligent approaches are required to address these challenges effectively [6,7]. Recent developments in artificial intelligence and ML have significantly improved the capability to detect and prevent malicious activities in vehicular networks. These techniques allow systems to learn complex patterns from data and distinguish between normal and abnormal behavior with higher accuracy [8]. However, the rapid growth of connected vehicles and the adoption of advanced communication technologies like 5G have expanded the potential attack surface. This has increased the urgency for robust and scalable detection mechanisms to ensure secure communication.

Furthermore, the widespread use of wireless communication in vehicular environments makes it easier for attackers to exploit vulnerabilities and launch coordinated attacks. Although several detection approaches have been proposed, many existing systems still face limitations in scalability, flexibility, and real-time performance [9]. Therefore, there is a critical need for advanced solutions capable of efficiently identifying malicious behavior while maintaining the reliability and efficiency of vehicular communication systems [10].

2. Literature Survey

Rashid, et al. [11] attacked by malicious nodes, especially DDoS attack detection. Several solutions are presented to overcome the issue, but none are solved in a real-time scenario using ML. They proposed a distributed multi-layer classifier and evaluated the results using OMNET++ and SUMO with ML classification using GBT, LR, MLPC, RF, and SVM models. Kaur, et al. [12] uses RaeSE and OMNET++ under the attack development using the web servers. Simulated DDoS attacks were generated using random access to the network effects. In this technique, multiple routers and factors are involved in addressing the issue raised in this technique. The network access ratio and packet drop ratio are observed in this technique.

D'Angelo, et al. [13] developed an algorithm that categorizes CAN bus messages in real-time as legitimate or illegitimate, alerting when malicious messages are detected. Through analysis of in-vehicle networks, researchers have identified features, like device fingerprints, clock offsets, and frequency characteristics, that can enhance detection. Khan, et al. [14] introduced an IDS for CAN networks that uses thresholds and statistical attack characteristics to improve detection efficiency. In related literature, an RNN-LSTM classifier was used to construct ECU fingerprint signals based on frequency domain features, effectively identifying flooding attacks and offering new protection strategies. Wang, et al. [15] proposed a method combining CNNs with anomaly detection and Kalman filtering to detect anomalous messages in vehicles. Another approach involves a real-time system that constructs a timing model by monitoring CAN bus traffic, identifying anomalies without predefined specifications.

Hyun Min Song, et al. [16] developed an IDS using a deep convolutional neural network (DCNN) to protect the CAN bus. These methods can handle large-scale data and complex features but require significant computational resources and labeled data for training and often lack interpretability. Deep belief networks (DBN) can pre-train feature extraction layers unsupervised, but their training is time-consuming. Zhou, X. et al. [17] Proposed the threat of cyberattacks on smart connected vehicles, numerous IDS have emerged. However, existing IDSs often prioritize enhancing detection accuracy while overlooking the time needed for training and detection. Moreover, they may not fully leverage the combined utilization of CAN bus IDs and the data field with external network data. Omar, H.A., et al. [18] proposed a method using RFID technology, which is implemented by verifying the vehicle's interactions with the cloud while driving on the road. According to the statistics, the vehicle will use electronic tags to send the data requiring authentication to the cloud storage.



Chen Wei, et al. [19] proposed a vehicle privacy protection method that combines the certificates issued to a vehicle by a trusted agency and pseudonyms authorized by a trusted agency to build an intelligent privacy authentication system to jointly authenticate a vehicle's identity. However, due to the frequent communication problems on the Internet of vehicles, the certification requirements cannot be met efficiently and in a timely manner. Wagner M, et al. [20] proposed a new blockchain architecture with local, physically verified transactions. With this new architecture, they proposed a protocol that protects vehicle self-organized networks (VANET) that do not regularly communicate with the RSU or other infrastructure components.

Rekha Gangula et al. [20] proposed a stacked autoencoder-based intrusion detection framework for Internet of Things (IoT) environments. The model extracted hierarchical deep features from network traffic data. A weighted loss function improved class imbalance handling and enhanced detection accuracy. Rekha Gangula et al. [21] proposed an IoT intrusion detection system using an Enhanced Flower Pollination Algorithm with an ensemble classifier. The optimization technique selected optimal features and parameters. The ensemble model improved classification robustness and reduced false alarms. K. Shailaja et al. [22] proposed an intelligent deep feature-based intrusion detection system for network applications. The framework utilized deep learning for feature extraction and classification. The system improved detection accuracy across multiple attack categories.

3. Proposed System

The proposed methodology as shown in Figure 1 presents a structured and intelligent framework for detecting malicious nodes in high mobility vehicular networks using a data-driven approach. The study integrates data preprocessing, feature extraction, and multi-model classification to handle the dynamic and complex nature of vehicular communication environments.

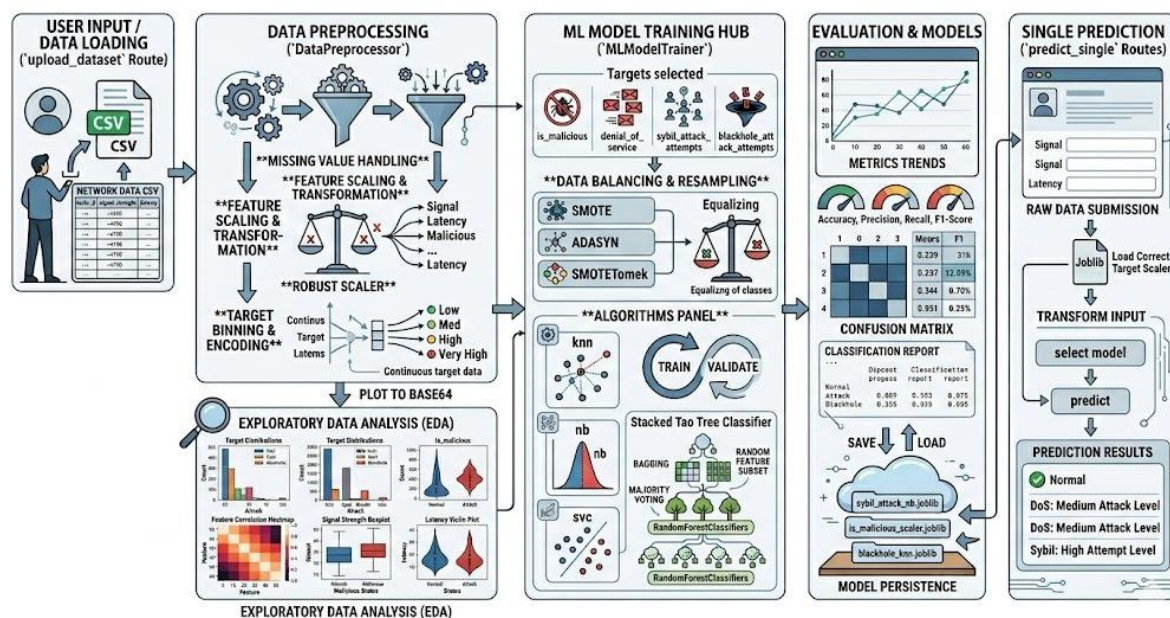


Fig. 1: Proposed System architecture.

The workflow begins with data acquisition from network activities, followed by cleaning, transformation, and scaling to ensure consistency and reliability. Multiple ML models are employed to analyse network behaviour and classify different types of malicious activities. A hybrid stacking-based approach enhances prediction accuracy by combining the strengths of individual models, enabling robust detection even in highly dynamic conditions. The system is supported by a lightweight storage



mechanism for efficient data handling, while a web-based interface enables real-time interaction, visualization, and monitoring. Continuous evaluation and retraining mechanisms further improve adaptability, ensuring consistent performance in evolving network scenarios.

User Interface (Web Browser)

- The user interacts with the system through a browser-based graphical interface, designed to simplify complex security operations for network administrators.
- Users can perform end-to-end actions, including dataset uploads, viewing EDA visualizations, triggering model training, and executing single-instance predictions.
- The dashboard dynamically displays high-fidelity analytical graphs, real-time performance metrics, and granular prediction results.
- All user interactions are converted into standardized HTTP requests and transmitted to the Flask web server for backend processing.

Flask Web Server (app.py)

- The Flask backend serves as the operational heart of the system, receiving user requests and routing them to specialized logic functions.
- It manages the orchestration of dataset uploads, preprocessing triggers, model training cycles, and real-time prediction workflows.
- The server acts as a coordinator, ensuring synchronized communication between the frontend interface, the data refinement modules, and the ML engine.
- It also handles the secure management of local storage, facilitating the quick retrieval of trained models and historical results.

Local Storage (Models & Results)

- The system maintains dedicated local directories for storing serialized models, feature scalers, and historical performance data.
- Model files are persisted using the joblib library, allowing for immediate "warm-start" loading during the prediction phase.
- By maintaining detailed performance metrics and output logs, the system enables longitudinal evaluation and comparative analysis across different training runs.
- This persistent storage layer eliminates the need for redundant retraining, significantly reducing the system's computational footprint.

Raw Data (CSV Input)

- The dataset, provided in CSV format, acts as the primary telemetry source for the security analysis engine.
- It captures critical network-related features, including signal strength, latency, packet delivery ratios, and various communication parameters.
- The data structure includes multiple target variables, allowing the system to perform multi-label classification for diverse attack types.
- This raw telemetry is forwarded directly to the preprocessing module, where it undergoes structural refinement and noise reduction.



Data Preprocessing & Feature Extraction (data_preprocessing.py)

- The dataset is subjected to a rigorous cleaning process that includes the handling of missing values and the systematic selection of high-impact features.
- Numeric features are transformed using RobustScaler, which utilizes the interquartile range to normalize data while effectively neutralizing the influence of extreme outliers.
- Irrelevant identifiers and redundant columns are stripped away to prevent model overfitting and improve overall processing speed.
- The final output of this module is a structured, numerical feature set optimized for high-dimensional ML analysis.

Data Balancing Module

- To address the inherent scarcity of attack data, the system applies a suite of advanced balancing techniques, including SMOTE, ADASYN, and SMOTETomek.
- These algorithms generate synthetic samples for minority classes (attacks) to ensure the model does not become biased toward "Normal" network behavior.
- SMOTETomek, specifically, cleans the overlapping class boundaries by removing "Tomek links" after oversampling, resulting in a clearer decision space.
- This module is essential for improving the system's sensitivity (recall) toward rare but high-impact security breaches.

ML Models (KNN, SVC, GNB, STT)

- The balanced dataset is fed into a diverse committee of classifiers to establish a robust detection baseline.
- KNN provides instance-based learning by evaluating distance metrics, while SVC works to identify the optimal hyperplanes for separating malicious from benign traffic.
- GNB applies probabilistic classification, which is particularly effective for features following a normal distribution.
- The STT model acts as a specialized hybrid ensemble, combining multiple decision tree architectures to achieve superior accuracy and stability over individual learners.

Model Training & Evaluation

- The refined dataset is split into training and testing subsets to facilitate a rigorous validation process.
- Each model is trained on the balanced dataset and subsequently challenged with unseen testing data to measure its real-world effectiveness.
- The system calculates a comprehensive suite of metrics, including Accuracy, Precision, Recall, and F1-score, to provide a 360-degree view of model performance.
- These results are compared in real-time, allowing the system to highlight the most effective model for specific network environments.

Prediction Results & Target Output

- The trained architecture generates simultaneous predictions for three critical security target variables:



- **Denial of Service (DoS):** Identifying attempts to overwhelm network resources.
- **Sybil Attack Attempts:** Detecting the creation of multiple fake identities to subvert system reputation.
- **Blackhole Attack Attempts:** Pinpointing nodes that maliciously drop incoming packets.
- Results are rendered in the UI, providing administrators with the specific attack type and an assessment of activity severity.

Model Retraining Mechanism

- The framework includes an "Evolutionary Module" that allows for the seamless retraining of models as new network data and threat signatures are captured.
- Updated datasets are automatically routed through the established preprocessing and balancing pipeline to maintain consistency.
- By adapting to new patterns, the system prevents "model decay" and remains effective against zero-day or evolving attack strategies.
- This ensures that the security infrastructure remains a "living" system that grows more intelligent with every bite of network traffic it analyzes.

3.1 Flask

Flask is a lightweight and flexible Python web framework designed to help developers build scalable and modular web applications. It follows a micro-framework architecture, meaning it provides the core tools needed for web development while allowing developers to extend its functionalities with external libraries as needed. Flask is known for its simplicity, ease of use, and clean design, making it ideal for integrating ML models, visualizations, and data processing workflows into fully functional web interfaces. In this research, Flask serves as the backbone of the entire ML pipeline by connecting the frontend user interface with backend processing logic. It efficiently handles user interactions, receives uploaded datasets, triggers preprocessing functions, manages ML model training, and generates predictions in real time. Flask's routing, templating, request handling, and data transfer capabilities ensure smooth communication between the ML modules and the user interface. Its modular structure also enables scalability and easy deployment on cloud platforms.

Routing and URL Mapping: Flask's `@app.route` decorator is used to define different endpoints such as `/upload dataset`, `/eda`, `/train models`, `/performance`, and `/predict single`. Each route handles a specific part of the workflow, including dataset upload, visualization display, model training, and prediction generation. This routing mechanism ensures that every user action corresponds to a clearly defined backend process.

Template Rendering: The research uses Flask's `render_template()` function to load and display HTML pages such as `home.html`, `eda.html`, `performance.html`, and `predict_single.html`. This helps separate the frontend (HTML/CSS/JS) from the backend logic, making the application organized and user-friendly. Templates are dynamically filled with plots, metrics, and prediction results.

File Upload Handling: Using Flask's `request.files` and `secure_filename()`, the application accepts user-uploaded CSV datasets and securely stores them in the uploads directory. Flask ensures safe file handling, preventing malicious uploads and enabling smooth dataset processing.



Form and JSON Data Processing: Flask's request. Form and request.get_json() features process user inputs for model selection and prediction. This allows the system to collect numeric values, algorithm choices, and other settings directly from the interface and feed them into the ML modules.

Flash Messaging System: The flash() function is used to send notifications and status messages to the user, such as successful dataset uploads, errors, or model training statuses. It improves the user experience by giving immediate feedback on interactions.

Sending Files and Images: The application uses send file() and base64 encoded images to display EDA plots dynamically on the webpage. The plot_to_base64() function converts matplotlib graphs into browser-friendly images, which are rendered directly in HTML templates.

Integration with Backend ML Modules: Flask acts as the communication layer between user actions and backend components like Data Preprocessor and ML Model Trainer. When a user uploads a dataset or triggers model training, Flask passes the data to these modules, retrieves outputs, and sends results back to the interface.

4. Results and Discussion

Fig. 2 illustrates the distribution of target variables used for detecting malicious activities in the vehicular network dataset. The figure presents multiple plots showing the frequency of different attack categories, including denial of service, sybil attack attempts, and blackhole attack attempts. It depicts how the data is distributed across various severity levels, providing insight into class balance and variation among attack types. The visualization highlights the presence of multiple classes for each target, indicating the complexity of the classification problem. It also helps in identifying whether the dataset is balanced or skewed across different attack levels.

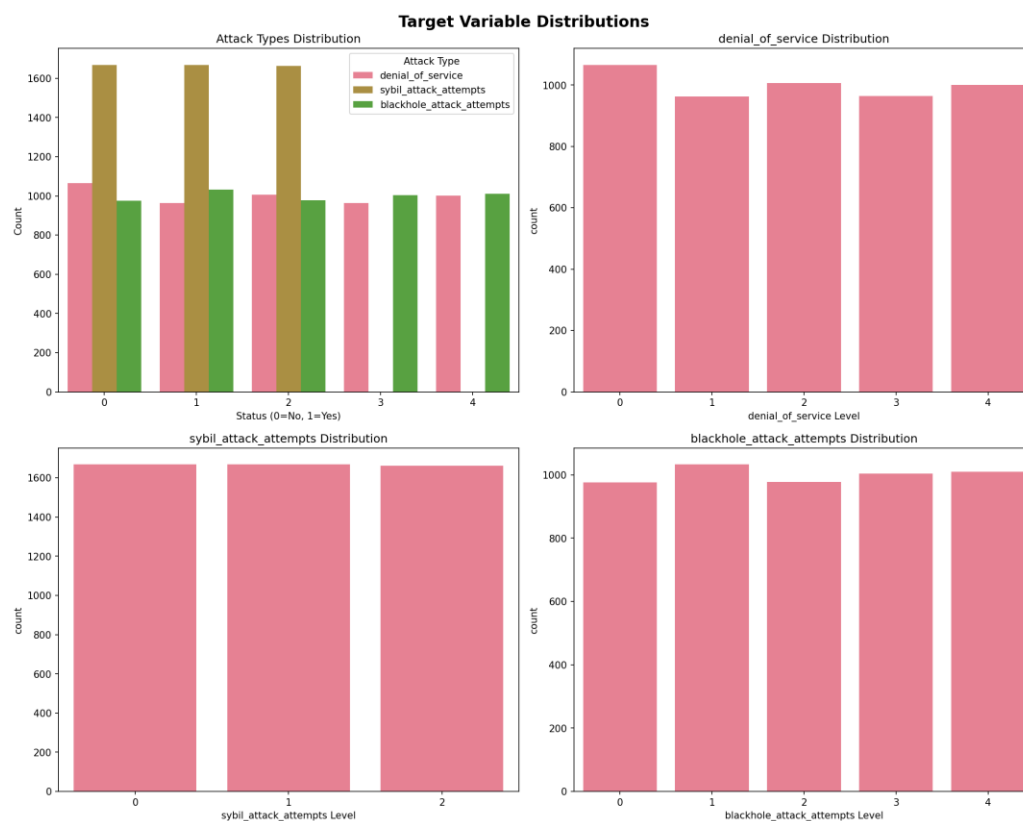


Fig. 2: Target variable distributions.



Fig. 3 illustrates the feature correlation heatmap representing the relationships between different variables in the vehicular network dataset. The figure depicts the strength and direction of correlations among features such as packet transmission metrics, latency, signal strength, and trust-related parameters. It highlights strong positive correlations between `trust_score`, `neighbor_trust_score_avg`, and `historical_trust_score`, indicating their close interdependence. Most other features show weak correlations, suggesting that they contribute independently to the model without redundancy. The visualization helps in identifying important feature interactions and potential multicollinearity within the dataset.

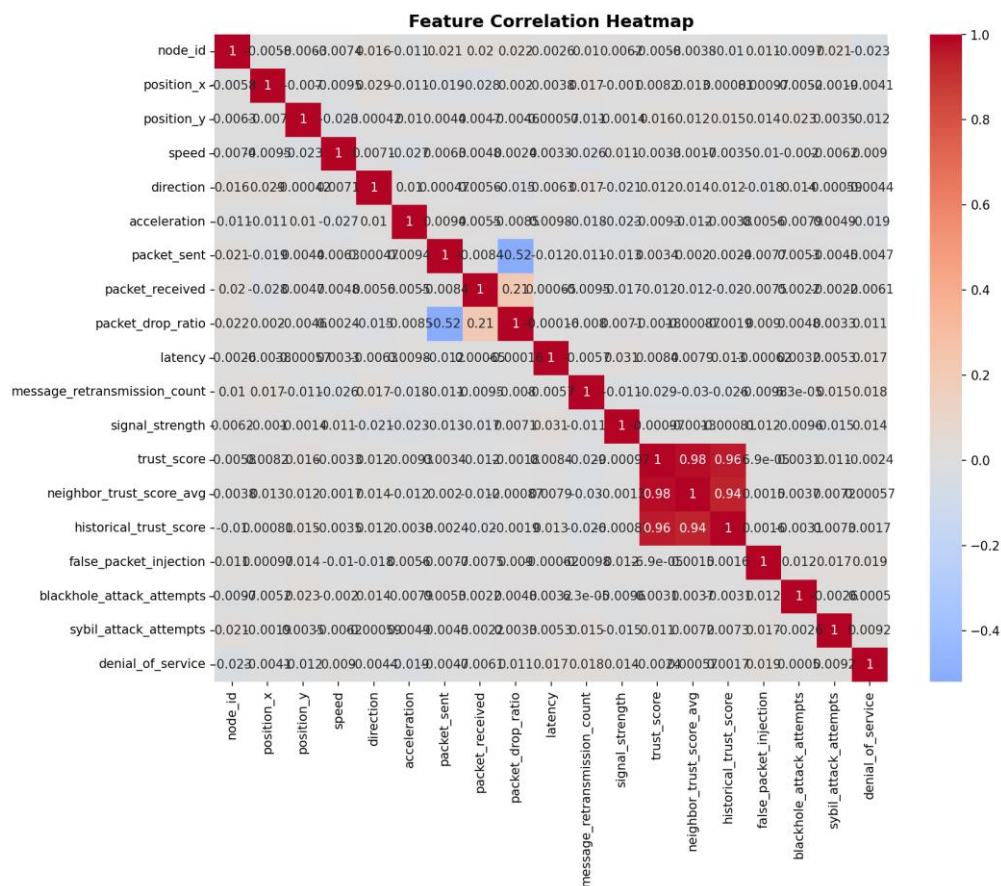


Fig. 3: Feature correlation heatmap.

Fig. 4(a) illustrates the model performance comparison for denial-of-service classification results using multiple ML algorithms. The figure presents evaluation metrics such as accuracy, precision, recall, and F1-score for KNN, SVC, GNB, and STT models. The accuracy achieved by KNN is 26.20%, while SVC shows a slightly higher accuracy of 29.00%. The GNB model records an accuracy of 26.20%, indicating comparable performance to KNN. In contrast, the STT model significantly outperforms all other models with an accuracy of 100.00%. This highlights the effectiveness of the ensemble-based STT approach in handling complex attack patterns.

Fig. 4(b) depicts the model performance comparison for sybil attack attempts classification across different ML models. The figure shows that KNN achieves an accuracy of 35.70%, while SVC records an accuracy of 33.80%. The GNB model performs similarly with an accuracy of 35.60%, indicating consistent results among baseline models. The STT model again demonstrates outstanding performance with an accuracy of 99.90%, far exceeding other algorithms. This indicates that the ensemble approach effectively captures patterns related to sybil attacks. The comparison highlights the limitations of individual models and the advantage of hybrid learning techniques.



Fig. 4(c) illustrates the model performance comparison for blackhole attack attempts classification using multiple algorithms. The figure shows that KNN achieves an accuracy of 19.90%, while SVC records a slightly lower accuracy of 19.40%. The GNB model performs relatively better among baseline models with an accuracy of 21.70%. However, the STT model significantly outperforms all other models by achieving an accuracy of 100.00%. This demonstrates the capability of the ensemble model to handle complex and imbalanced data effectively. The comparison clearly indicates the superiority of STT over traditional ML models.

Denial Of Service Classification Results				
Algorithm	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)
K-Nearest Neighbors	26.20%	29.14%	26.20%	26.70%
Support Vector Classifier	29.00%	29.81%	29.00%	29.27%
Naive Bayes	26.20%	28.58%	26.20%	26.53%
Stacked Tao Tree classifier	100.00%	100.00%	100.00%	100.00%

(a)

Sybil Attack Attempts Classification Results				
Algorithm	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)
K-Nearest Neighbors	35.70%	35.39%	35.70%	34.65%
Support Vector Classifier	33.80%	33.84%	33.80%	33.80%
Naive Bayes	35.60%	35.93%	35.60%	35.05%
Stacked Tao Tree classifier	99.90%	99.90%	99.90%	99.90%

(b)

Blackhole Attack Attempts Classification Results				
Algorithm	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)
K-Nearest Neighbors	19.90%	19.80%	19.90%	19.43%
Support Vector Classifier	19.40%	19.51%	19.40%	19.37%
Naive Bayes	21.70%	21.66%	21.70%	20.08%
Stacked Tao Tree classifier	100.00%	100.00%	100.00%	100.00%

(c)

Fig. 4: Model Performance Comparison of Various Methods. (a) Denial of Service, (b) Sybil Attack Attempts. (c) Blackhole Attack Attempts

5. Conclusion

The study presents an effective and intelligent approach for detecting malicious nodes in high mobility vehicular networks using ML techniques. The system successfully integrates data preprocessing, exploratory analysis, and multi-model classification to handle complex network behaviour. The performance evaluation demonstrates that traditional models such as KNN, SVC, and GNB show limited accuracy due to data complexity and class imbalance. In contrast, the STT model achieves significantly improved results, reaching near-perfect accuracy across all target variables. The use of



advanced data balancing techniques further enhances model performance and reduces bias toward majority classes. The system can handle multi-target classification efficiently, providing reliable detection of denial of service, sybil, and blackhole attacks. The integration of a web-based interface allows real-time interaction and prediction, improving usability and practical implementation. The system shows strong performance improvements in terms of accuracy, robustness, and scalability. This demonstrates the effectiveness of ensemble-based learning in addressing complex network security challenges. Future scope includes extending the system with deep learning models to further enhance detection accuracy and adaptability. It can also be integrated with real-time vehicular communication systems for live monitoring and automated threat prevention.

References

- [1] Rashid, K.; Saeed, Y.; Ali, A.; Jamil, F.; Alkanhel, R.; Muthanna, A. An Adaptive Real-Time Malicious Node Detection Framework Using ML in Vehicular Ad-Hoc Networks (VANETs). *Sensors* 2023, 23, 2594. <https://doi.org/10.3390/s23052594>
- [2] Al-Omaisi, H.; Sundararajan, E.A.; Abdullah, N.F. Towards vanet-ndn: A framework for an efficient data dissemination design scheme. In *Proceedings of the 2019 International Conference on Electrical Engineering and Informatics (ICEEI)*, Bandung, Indonesia, 9–10 July 2019; pp. 412–417.
- [3] Pournaghi, S.M.; Zahednejad, B.; Bayat, M.; Farjami, Y. NECPPA: A novel and efficient conditional privacy-preserving authentication scheme for VANET. *Comput. Networks* 2018, 134, 78–92.
- [4] Lee, M.; Atkison, T. Vanet applications: Past, present, and future. *Veh. Commun.* 2021, 28, 100310.
- [5] Sharma, A.; Pandey, K. Recent Advancements in Techniques Used to Solve the RSU Deployment Problem in VANETs: A Comprehensive Survey. *Int. J. Sens. Wirel. Commun. Control* 2022, 12, 184–193.
- [6] Ganesh, A.; Ayyasamy, S. Enhanced Approach in VANETs for Avoidance of Collision with Reinforcement Learning Strategy. In *Proceedings of the International Conference on Artificial Intelligence for Smart Community*, Perak, Malaysia, 17–18 December 2020; Springer: Berlin/Heidelberg, Germany, 2022; pp. 419–428.
- [7] Pandey, P.K.; Kansal, V.; Swaroop, A. VANETs: Architecture, challenges, and applications. In *Handling Priority Inversion in Time-Constrained Distributed Databases*; IGI Global: Hershey, PA, USA, 2020; pp. 224–239.
- [8] Ali, I.; Hassan, A.; Li, F. Authentication and privacy schemes for vehicular ad hoc networks (VANETs): A survey. *Veh. Commun.* 2019, 16, 45–61.
- [9] Mansoor, S.; Shahid, S.; Ashiq, K.; Alwada, N.; Javed, M.; Iqbal, S.; Ibrahim, H.A. Controlled growth of nanocomposite thin layer based on Zn-Doped MgO nanoparticles through Sol-Gel technique for biosensor applications. *Inorg. Chem. Commun.* 2022, 142, 109702.
- [10] Gaurav, A.; Gupta, B.B.; Peñalvo, F.J.G.; Nedjah, N.; Psannis, K. Ddos attack detection in vehicular ad-hoc network (vanet) for 5g networks. In *Security and Privacy Preserving for IoT and 5G Networks: Techniques, Challenges, and New Directions*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 263–278.



- [11] Grover, J. Security of Vehicular Ad Hoc Networks using blockchain: A comprehensive review. *Veh. Commun.* 2022, 34, 100458.
- [12] Raza, A.; Bukhari, S.H.R.; Aadil, F.; Iqbal, Z. An UAV-assisted VANET architecture for intelligent transportation system in smart cities. *Int. J. Distrib. Sens. Netw.* 2021, 17, 15501477211031750.
- [13] Reddy, S. K. R. Developing a Modular AI Framework to Enhance Scalability and Personalization in Next-Generation Reward Platforms.
- [14] Kalae, U. K. (2021). Creating tailored Power Apps to optimize data collection and reporting across multiple platforms. *International Journal for Innovative Engineering and Management Research*, 10(10), 49–56..
- [15] Purmani, S. S. R. (2025). Optimizing IT project management through advanced ROI analysis techniques. *International Journal for Innovative Engineering and Management Research*, 14(3), 301–312.
- [16] Song, H.M.; Woo, J.; Kim, H.K. In-vehicle network intrusion detection using deep convolutional neural network. *Veh. Commun.* 2020, 21, 100198.
- [17] Abboud, K.; Omar, H.A.; Zhuang, W. Inter working of DSRC and Cellular Network Technologies for V2X Communications: A Survey. *IEEE Trans. Veh. Technol.* 2016, 65, 9457–9470.
- [18] Zhongmei, L.; Wei, C.; Jie, W.; Haitao, Y. Very low latency and high reliability communication of Internet of Vehicles: Status and Outlook. *Signal Process.* 2019, 35, 1773–1783.
- [19] Arushi, A.; Kumar, Y.S. Block Chain Based Security Mechanism for Internet of Vehicles (IoV). In *Proceedings of the 3rd International Conference on Internet of Things and Connected Technologies (ICIoTCT)*, Jaipur, India, 26–27 March 2018; pp. 267–272.
- [20] Wagner, M.; Mcmillin, B. Cyber-Physical Transactions: A Method for Securing VANETs with Blockchains. In *Proceedings of the IEEE Pacific Rim International, Symposium on Dependable Computing*, Taipei, Taiwan, 4–7 December 2018; pp. 64–73.
- [21] Rekha Gangula, V. Murali Mohan, Ranjeeth Kumar. M, “Stacked autoencoder with weighted loss function for intrusion detection in IoT application”. *Multimed Tools Appl* (2024). <https://doi.org/10.1007/s11042-024-19962-7>.
- [22] Rekha Gangula, V. Murali Mohan, Ranjeeth Kumar. M, “Network Intrusion Detection System for Internet of Things based on Enhanced Flower Pollination Algorithm and Ensemble Classifier”, *Concurrency and Computation: Practice and Experience – Wiley Online Library*, Accepted on 05 April 2022, ISSN: 1532-0634 (online), ISSN: 1532-0626(print), Impact Factor: 1.536. Standard age of Journal - 21 Yrs. (since 2001).
- [23] K Shailaja, B Srinivasulu, Lingala Thirupathi, Rekha Gangula, Thejoramnaresh Reddy Boya, Vemulamma Polem ,”An Intelligent Deep Feature Based Intrusion Detection System for Network Applications”. *Wireless Pers Commun* 129, 345–370 (2023). <https://doi.org/10.1007/s11277-022-10100-w>